Parameterized Algorithms for Zero Extension and Metric Labelling Problems

Felix Reidl
Royal Holloway, University of London, TW20 0EX, UK
Felix.Reidl@rhul.ac.uk

Magnus Wahlström
Royal Holloway, University of London, TW20 0EX, UK
Magnus.Wahlstrom@rhul.ac.uk

Abstract

We consider the problems Zero Extension and Metric Labelling under the paradigm of parameterized complexity. These are natural, well-studied problems with important applications, but have previously not received much attention from this area.

Depending on the chosen cost function μ, we find that different algorithmic approaches can be applied to design FPT-algorithms: for arbitrary μ we parameterize by the number of edges that cross the cut (not the cost) and show how to solve Zero Extension in time $O(|D|^{O(k^2)}n^4 \log n)$ using randomized contractions. We improve this running time with respect to both parameter and input size to $O(|D|^{O(k)}m)$ in the case where μ is a metric. We further show that the problem admits a polynomial *sparsifier*, that is, a kernel of size $O(k^{|D|+1})$ that is *independent of the metric* μ.

With the stronger condition that μ is described by the distances of leaves in a tree, we parameterize by a *gap parameter* $(q-p)$ between the cost of a true solution q and a 'discrete relaxation' p and achieve a running time of $O(|D|^{q-p}|T|m + |T|\phi(n,m))$ where T is the size of the tree over which μ is defined and $\phi(n,m)$ is the running time of a max-flow computation. We achieve a similar result for the more general Metric Labelling, while also allowing μ to be the distance metric between an arbitrary subset of nodes in a tree using tools from the theory of VCSPs. We expect the methods used in the latter result to have further applications.

2012 ACM Subject Classification Theory of computation → Fixed parameter tractability

Keywords and phrases FPT, VCSP, cut problem, gap parameter

Digital Object Identifier 10.4230/LIPIcs.ICALP.2018.94

Related Version A full version of the paper is available at [27], `http://arxiv.org/abs/1802.06026`.

Funding Supported by EPSRC grant EP/P007228/1

1 Introduction

The task of extending a partial labelling of data points to a full data set while minimizing an error function is a natural step for many scientific and engineering tasks. For the particular case of data imposed with a binary relationship, we find the problems Zero Extension and Metric Labelling to be well-suited for optimization in image processing [1], social network classification [24], or sentiment analysis [25]. The problems are as follow. For Zero Extension, we are given a graph G and a partial labelling $\tau\colon S \to D$, for terminals $S \subseteq V(G)$, and a cost function $\mu\colon D \times D \to \mathbb{R}^+$. Our task is to compute a labelling $\lambda\colon V(G) \to D$ which agrees with τ on S, subject to the following cost: for each edge $uv \in G$ we pay $\mu(\lambda(u), \lambda(v))$. In Metric Labelling, we are given G, μ as above, and a *labelling cost* $\sigma\colon V(G) \times D \to \mathbb{R}^+$.

45th International Colloquium on Automata, Languages, and Programming (ICALP 2018).
Editors: Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella;
Article No. 94; pp. 94:1–94:14
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Again we are asked to compute a labelling λ and in addition to the above edge-costs we now also pay $\sigma(v, \lambda(v))$. This model allows us to emulate terminals by making the cost $\sigma(v, \lambda(v))$ prohibitive for all but the required label $\lambda(v)$. Both problems are generalizations of MULTIWAY CUT (we simply let μ be identically one for all distinct pairs), which has garnered considerable attention from the FPT community and formed a crystallization nucleus for the very fruitful research of cut-based problems (see e.g. [20, 17, 3, 11, 22]).
We apply most of these tools in the following, but we wish to highlight the use of tools and relaxations from *Valued CSPs* (VCSPs) for designing FPT algorithms under gap parameters. VCSPs are a general framework for expressing optimisation problems, via the specification of a set Γ of *cost functions* (aka *constraint language*). Many important problems correspond to VCSP for a specific language, including every choice of a specific metric for the problems above. Thapper and Živný [30] characterized the languages Γ for which the resulting VCSP is tractable.

The use of a tractable VCSP as a *discrete relaxation* of an NP-hard optimisation problem has led to powerful FPT algorithms [11] (see also related improvements [12, 32]). In this paper, we advance this research in two ways. First, previous approaches have required the relaxation to have a *persistence* property, which allows an optimum to be found by sequentially fixing variables. Here, we relax this condition to a weaker *domain consistency* property. Second, we use a folklore result from VCSP research to restrict the behaviour of an instance's optimal solutions in order to facilitate the proof that the domain consistency property holds for the relevant VCSPs. See Section 5 for details.

Related work. ZERO EXTENSION and METRIC LABELLING have been researched primarily from the perspective of efficient and approximation algorithms (see [19] for an overview and hardness results). Kleinberg and Tardos [29] introduced METRIC LABELLING and provided a $O(\log |S| \log\log |S|)$ approximation. A result by Fakcharoenphol *et al.* regarding embedding general metrics into tree metrics [7] improves the ratio of this algorithm to $O(\log |S|)$ and a lower bound of $O((\log |S|)^{1/2-\varepsilon})$ was proved by Chuzhoy and Naor [4]. Karzanov [14] introduced ZERO EXTENSION with the specific case of μ being a graph metric, that is, equal to the distance metric of some graph H. His central question—for which graphs H the problem is tractable—was recently fully answered by Hirai [9]. Picard and Ratliff earlier showed that an equivalent problem is tractable on trees [26]. Fakcharoenphol *et al.* showed that the problem can be approximated to within a factor of $O(\log |S| / \log\log |S|)$ [6]. Karloff *et al.* used the approach by Chuzhoy and Naor to show that no factor of $O((\log |S|)^{1/4-\varepsilon})$ for any $\varepsilon > 0$ is possible unless $\mathsf{NP} \subseteq \mathsf{QP}$ [13]. More recently, Hirai and Pap [8, 10] studied the problem from a more structural angle and we make use of their duality result.

Our results. We study both problems from the perspective of parameterized complexity. As the choice of metric has a strong effect on the problem complexity, we give a range of results, from the more generally applicable to the algorithmically stronger, both in terms of running time and parameterization. When μ is a general cost function or a metric, we will parametrize not by the *cost* of a solution but by the number of *crossing edges*, i.e., bichromatic edges under a labelling λ. This lets us consider μ with zero-cost pairs. For general cost functions, we employ the technique of randomized contractions [3] and prove:

▶ **Theorem 1** ($\star$[1]). ZERO EXTENSION *can be solved in time $O(|D|^{O(k^2)} n^4 \log n)$ where k is a given upper bound on the number of crossing edges in the solution.*

[1] Results marked by $\star$ are found in the full version of the paper [27]

When μ is a metric, we are able to give a linear-time FPT algorithm, while also improving the dependency on the parameter, using important separators [20]:

▶ **Theorem 2.** Zero Extension *with metric cost functions can be solved in time* $O(|D|^{O(k)} \cdot m)$ *where* k *is a given upper bound on the number of crossing edges in the solution.*

For the general metric setting, we also have our most surprising result, demonstrating that Zero Extension admits a *sparsifier*; that is, we prove that it admits a polynomial kernel *independent of the metric* μ. This result crucially builds on the technique of *representative sets* [17, 21, 18]. The exact formulation of the result is somewhat technical and we defer it to Section 4.2, but roughly, we obtain a kernel of size $O(k^{|S|+1})$, independent of μ, where k is again the number of crossing edges. This result is a direct, seemingly far-reaching generalization of the polynomial kernel for s-Multiway Cut [17].

Next, we consider the case when $\mu\colon D \times D \to \mathbb{Z}^+$ is induced by the distance in a tree T with $D \subset V(T)$. Here, relaxing the problem to allow all labels $V(T)$ as vertex values defines a tractable discrete relaxation, in the sense discussed above. In particular, we can compute a *relaxed solution cost* p in polynomial time which lower-bounds the optimal integral solution q. Using techniques from VCSP, we design a gap-parameter algorithm:

▶ **Theorem 3.** *Let* $I = (G, \tau, \mu, q)$ *be an instance of* Zero Extension *where* μ *is an induced tree metric on a set of labels* D *in a tree* T*, and let* $\hat{I} = (G, \tau, \hat{\mu}, q)$ *be the relaxed instance. Let* $p = \mathbf{cost}(\hat{I})$*. Then we can solve* I *in time* $O(|D|^{q-p}|T||D|nm)$*.*

For the further restriction when μ corresponds to the distances of the leaves D of a tree T, we obtain an algorithm with a slightly better polynomial dependence. Moreover, it uses only elementary operations like computing cuts and flows:

▶ **Theorem 4.** *Let* $I = (G, \tau, \mu, q)$ *be an instance of* Zero Extension *where* μ *is a leaf metric on a set of labels* D *in a tree* T*, and let* $\hat{I} = (G, \tau, \hat{\mu}, q)$ *be the relaxed instance. Let* $p = \mathbf{cost}(\hat{I})$*. Then we can solve* I *in time* $O(|D|^{q-p}|T|m + |T|\phi(n, m))$*, where* ϕ *is the time needed to run a max-flow algorithm.*

Finally, we apply the VCSP toolkit to Metric Labelling and obtain a similar gap algorithm (see Section 5 for undefined terms).

▶ **Theorem 5.** *Let* $I = (G, \sigma, \mu, q)$ *be an instance of* Metric Labelling *where* μ *is an induced tree metric for a tree* T *and a set of nodes* $D \subseteq V(T)$*, and where every unary cost* $\sigma(v, \cdot)$ *admits an interpolation on* T*. Let* $\hat{I} = (G, \hat{\sigma}, \hat{\mu}, q)$ *be the relaxed instance, and let* $p = \mathbf{cost}(\hat{I})$*. Then the instance* I *can be solved in time* $O^*(|D|^{q-p})$*. In particular, this applies for any* σ *if* D *is the set of leaves of* T*.*

2 Preliminaries

For a graph $G = (V, E)$ we will use $n_G = |V|$ and $m_G = |E|$ to denote the number of vertices and edges, respectively. For two disjoint vertex sets $A, B \subseteq V$ we write $E(A, B)$ to denote the edges that have one endpoint in A and the other in B. We write d_G for the distance-metric induced by G, that is, $\mathrm{d}_G(u, v)$ is the length of a shortest path between vertices $u, v \in V(G)$. We denote by $N_G(v)$ and $N_G[v]$ the open and closed neighbourhood of a vertex. For a vertex set $S \subseteq V(G)$ we write $\delta_G(S)$ to denote the set of edges with exactly one endpoint in S. We omit the subscript G, if clear from the context, in all these notations.

Let T be a tree and $xy \in T$ an edge, then we use the notation T_x to denote the component of $T - xy$ that contains x. We call a sequence of nodes $x_1 x_2 \dots x_p$ in T a *monotone sequence* if $x_1 \leqslant_P x_2 \leqslant_P \ldots \leqslant_P x_p$ where P is a path in T and $\leqslant_P$ is the linear order induced by P. Note that $x_i = x_{i+1}$ is explicitly allowed. For two nodes $x, y \in T$ we will denote the unique x-y-path in T by $T[x, y]$. For a vertex set S, an *S-path packing* is a collection of edge-disjoint paths $\mathcal{P}$ that connect pairs of vertices in S. We will also consider *half-integral* path packings where every edge is allowed to be used by at most two paths.

Let D be a set of labels. For a graph G we call a function $\tau \colon S \to D$ for $S \subseteq V(G)$ a *partial labelling* and a function $\lambda \colon V(G) \to D$ a *labelling*. The labelling λ is an *extension* of τ if λ and τ agree on S, that is, for every vertex $u \in S$ we have that $\lambda(u) = \tau(u)$. Given a graph G and a labelling λ we call an edge $uv \in E(G)$ *crossing* if $\lambda(u) \neq \lambda(v)$. A *$\tau$-path packing* is a collection $\mathcal{P}$ of edge-disjoint paths such that every path $P \in \mathcal{P}$ connects two vertices that receive distinct labels under τ (and both are labelled).

A *cost function* over D is a symmetric positive function $\mu \colon D \times D \to \mathbb{R}^+$. We call it *simple* if $\mu(x, x) = 0$. A cost function is a *metric* if it is simple and further obeys the triangle inequality; it is a *tree metric* if it corresponds to the distance metric of a tree. We derive an *induced tree metric* from a tree metric by restricting its domain to a subset D of the nodes of the underlying tree. A *leaf metric* is an induced tree metric where D is the set of leaves of the tree. Given a cost function μ, we define the cost of a labelling λ of a graph G as $\mathbf{cost}_\mu(\lambda, G) = \sum_{uv \in G} \mu(\lambda(u), \lambda(v))$.

3 Cost functions: Randomized Contractions

We apply the framework by Chitnis *et al.* [3] to show that the general case of Zero Extension is in FPT when parameterized by the number of crossing edges. Note that crossing edges could incur an arbitrary cost, including zero. The stronger parameterization of only counting the number of crossing edges at non-zero cost makes for an intractable problem: With zero-cost edges, we can express the problem H-Retraction for reflexive graphs H, which asks us to find a retraction of a graph G into a fixed graph H. This problem is already NP-complete for H being the reflexive 4-cycle [31] and thus Zero Extension is paraNP-complete for $k = 0$ when parameterized by the number of non-zero crossing edges or the total cost.

A *(σ, κ)-good* separation is a partition (L, R) of $V(G)$ such that $|L|, |R| > \sigma$, $|E(L, R)| \leqslant \kappa$, and both $G[L]$ and $G[R]$ are connected. There exists an algorithm that finds a (σ, κ)-good separation in time $O((\sigma+\kappa)^{O(\min(\sigma,\kappa))} n^3 \log n)$ (Lemma 2.2 in [3]) or concludes that the graph is *(σ, κ)-connected*, that is, no such separation exists. The following lemma is a slight reformulation of Lemma 1.1 in [3] which in turn is based on splitters as defined by Naor *et al.* [23]:

▶ **Lemma 6** (Edge splitter). *Given a set E of size m and integers $0 \leqslant a, b \leqslant m$ one can in time $O((a+b)^{O(\min\{a,b\})} m \log m)$ construct a set family $\mathcal{F}$ over E of size at most $O((a+b)^{O(\min\{a,b\})} \log m)$ with the following property: for any disjoint sets $A, B \subseteq E$ with $|A| \leqslant a$ and $|B| \leqslant b$ there exists a set $H \in \mathcal{F}$ with $A \subseteq H$ and $B \cap H = \emptyset$.*

We first prove that Zero Extension can be solved on such highly connected instances and then apply the 'recursive understanding' framework to handle graphs with good separations.

▶ **Lemma 7.** *Let G be (σ, k)-connected for some $\sigma > k$. Then we can find an optimal solution in time $O((|D| + 2\sigma k + k)^{O(k)} (n + m) \log n)$.*

Proof sketch. Let $\lambda \in \mathbf{opt}(I)$ and let E_λ be the crossing edges with endpoints $V(E_\lambda)$. Let $C_0, C_1, \ldots, C_\ell$ be the connected components of $G - E_\lambda$ with C_0 being the largest one. Since G is (σ, k)-connected, we know that $\ell \leqslant k$ and that all components $C_1, \ldots, C_\ell$ have size at most σ (*cf.* Lemma 3.6 in [3]). We will assume that $|C_0| > \sigma$, otherwise $|V(G)| \leqslant \sigma k$ vertices and we find E_λ by brute-force.

We proceed by colouring $E(G)$. Such a colouring is *successful* if 1) E_λ is red, 2) each C_i, $i \geqslant 1$, contains a blue spanning tree, and 3) each vertex $u \in C_0 \cap V(E_\lambda)$ is contained in a blue tree of size $\geqslant \sigma + 1$. It is easy to verify that we need to correctly colour a set $B \subseteq E(G)$, $|B| \leqslant (\sigma - 1)\ell + \sigma k \leqslant 2\sigma k$ edges blue while colouring a set $R \subseteq E(G)$, $|R| \leqslant k$, edges red. We construct an edge-splitter $\mathcal{F}$ with $a = 2\sigma k$ and $b = k$ according to Lemma 6 of size $O((2\sigma k + k)^{O(k)} \log m)$. By construction, at least one colouring in $\mathcal{F}$ will be successful.

Fix a successful colouring. Let G_B be the graph on the blue edges. Call a component of G_B *small* if it contains $\leqslant \sigma$ vertices and *big* otherwise. Our task is to recover $C_0, C_1, \ldots, C_\ell$. Every C_i, $i \geqslant 1$ is small in G_B and all components reachable from C_i via red edges must either be a solution component C_j, $j \geqslant 1$, *or* a big component in G_B. Thus, we can 'discover' the sets $C_1, \ldots, C_\ell$ by marking small components that contain a terminal and then successively mark small components with red edges into already marked components. Afterwards we identify the crossing edges E_λ and λ. The total running time to identify E_λ is $(2\sigma k + k)^{O(k)}(n + m) \log n$. Given E_λ, the final step is to find an optimal assignment. We simply try all possible assignments for non-terminal components in time $O(|D|^\ell k) = O(|D|^k k)$ and the claimed running time follows. ◀

With the well-connected cases handled, the theorem follows by a straightforward application of recursive understandings [3].

▶ **Theorem 1** ($\star$[2]). Zero Extension *can be solved in time $O(|D|^{O(k^2)} n^4 \log n)$ where k is a given upper bound on the number of crossing edges in the solution.*

4 General metrics: Pushing separators

We now consider the more restricted, but reasonable case that μ is a metric, observing the triangle inequality. We find that this allows a 'greedy' operation of *pushing* in a solution λ, which allows both the design of a faster algorithm (Section 4.1) and the computation of a *metric sparsifier* (Section 4.2). Throughout the section, let $I = (G = (V, E), \tau, \mu, q)$ be an instance of Zero Extension for an arbitrary metric μ, let S be the set of terminals of G, and let D be the set of labels. We assume that the following reductions have been performed on G: For every label ℓ used by τ there is a terminal t_ℓ, and every vertex v such that $\tau(v) = \ell$ has been identified with this terminal t_ℓ.

We first prove a useful lemma. Let $\lambda \colon V \to D$ be an extension of τ, and let $U = \lambda^{-1}(\ell)$ for some $\ell \in D$. By *pushing from ℓ in λ* we refer to the operation of relabelling vertices to grow the set U "as large as possible", without increasing the number of crossing edges. Formally, this refers to the following operation: Let C be the furthest min-cut between vertex sets U and $S - t_\ell$, respectively S if there is no terminal t_ℓ (cf. [20, Lemma 3]); let U' be the vertices reachable from U in $G - C$; and let λ' be the labelling where $\lambda'(v) = \ell$ for $v \in U'$ and $\lambda'(v) = \lambda(v)$ otherwise. Clearly, λ' is an extension of τ. We show that as long as μ is a metric (observing the triangle inequality), pushing does not increase the cost of the solution.

▶ **Lemma 8** (Pushing Lemma). *For any τ-extension λ and every label $\ell \in D$, pushing from ℓ in λ yields a τ-extension λ' with $\mathbf{cost}_\mu(\lambda', G) \leqslant \mathbf{cost}_\mu(\lambda, G)$.*

Proof sketch. Since C is a min-cut there is a set of paths that begin in $\delta(U)$, saturate C, and end in terminals $S - t_\ell$. By the triangle inequality, the cost incurred by λ on these paths is at least as large as that incurred by λ', and no other edge increases its cost in λ'. ◀

[2] Results marked by $\star$ are found in the full version of the paper [27]

An immediate consequence of the above lemma is the following reduction rule.

▶ **Corollary 9** (⋆). *We can reduce to the case where for every terminal $t_\ell \in S$, $\delta(t_\ell)$ is the unique $(t_\ell, S - t_\ell)$-min cut in G.*

4.1 An FPT algorithm

We now show that Zero Extension is FPT for a metric μ parameterized by $k + |D|$, where k is a bound on the number of crossing edges of an optimum λ. The algorithm uses Lemma 8 to guess a solution λ using the classical technique of *important separators*, pioneered by Marx [20] as an important technique for FPT algorithms solving cut problems. Our algorithm roughly follows the algorithm for Multiway Cut of Chen, Liu and Lu [2], with two complications. First, unlike in Multiway Cut, there may be crossing edges in λ that are not reachable by a terminal, which makes the branching more expensive; second, even after all crossing edges have been found, it still remains to find an optimal labelling λ.

Since these complications can be present or absent for different metrics μ, we describe the algorithm in stages, where the first stage identifies all crossing edges reachable from a terminal, the second stage identifies the remaining crossing edges, and the third stage finds an assignment λ. For specific metrics μ, it may then be possible to speed this up by skipping some steps. In summary, we show the following.

▶ **Theorem 2.** Zero Extension *with metric cost functions can be solved in time $O(|D|^{O(k)} \cdot m)$ where k is a given upper bound on the number of crossing edges in the solution.*

We begin by providing the running time for the first stage. This is analysed in terms of a lower bound p on the crossing number of any labelling λ. This may be defined as follows: First apply Corollary 9, then compute $p = \sum_{t \in S} |\delta(t)|/2$. It is known that p is a lower bound on the multiway cut number of (G, S) [28], hence also on the number of crossing edges of λ, making $k - p$ a valid gap parameter. (Note that p does not measure the *cost* of a crossing edge or path; such results are shown in Section 5.)

▶ **Lemma 10** (⋆). *Let p be the lower bound as above. In $O(4^{k-p}km)$ time and 4^{k-p} guesses, we can reduce to the case where every edge of $\delta(t)$ is a crossing edge in the optimal solution for every $t \in S$.*

A similar result (without a lower bound) finds the remaining crossing edges of a solution.

▶ **Lemma 11** (⋆). *Given an input from stage 1, with p edges already marked as crossing, in $O(4^{2k-p}m)$ time and 4^{2k-p} guesses we can reduce to the case where every edge of G is crossing in the optimal solution.*

After stage 2, the remaining graph contains at most k edges, hence at most $O(k)$ vertices, and it only remains to find the min-cost labelling of the non-terminal vertices. In the absence of any stronger structural properties of the metric μ, this last phase can be completed in $|D|^{O(k)}O(m)$ time. Theorem 2 follows.

4.2 A kernel for any metric

We next show that Zero Extension has a kernel of $O(k^{s+1})$ vertices for any metric μ, where k is a bound on the number of crossing edges of a solution and s is the number of labels of μ. Moreover, the kernel can be computed without access to μ. This gives us a kind of *metric sparsifier* for (G, S), up to parameters k and s, as follows. The result is an adaptation of the kernel for s-Multiway Cut of Kratsch and Wahlström [17]. For an instance I, let $\mathbf{cost}(I, k)$ be the minimum cost of a labelling with at most k crossing edges (otherwise ∞).

▶ **Theorem 12** (⋆). *Let $s \geqslant 3$ be a constant. For every graph $G = (V, E)$ with a set S of terminals, $|S| \leqslant s$, and integer k, there is a randomized polynomial-time computable set $Z \subseteq E$ with $|Z| = O(k^{s+1})$ such that for any instance $I = (G, \tau, \mu, q)$ of* Zero Extension *with S being the set of terminals in I and μ having at most s labels, if $\mathbf{cost}(I, k) < \infty$ then there is a τ-extension λ with crossing number at most k and cost $\mathbf{cost}(I, k)$ such that every crossing edge of λ is contained in Z.*

By contracting any edges not in $\bigcup_{t \in S} \delta(t) \cup Z$ we then get the kernelized instance (G', S).

5 Tree metrics: Gap algorithms and VCSP relaxations

In this section we present more powerful algorithms parameterized by the gap parameter for problems where the metric embeds into a tree metric. We begin by a purely combinatorial algorithm for Zero Extension on leaf metrics, then we move on to the more general Zero Extension and Metric Labelling problems for general induced tree metrics. The algorithms for the latter problems rely on the *domain consistency* property of the relaxation, which allows us to solve the problem by simply branching on the value of a single variable at a time. This property is shown by way of a detour into an analysis of properties of VCSP instances whose cost functions are *weakly tree submodular*, which is a tractable problem class containing tree metrics. The algorithms for these problems are then straight-forward.

At this point, we need to address a subtlety regarding the input cost function μ. So far, the cost function only had to obey basic properties that are easily verifiable or could be seen as a 'promise'. However, some of our arguments below will explictly need the tree T that induces the metric. Luckily this issue has been solved already: given a induced tree metric μ over D in matrix form, one can in time $O(|D|^2)$ compute a tree that induces μ [5]. If μ is a leaf metric, the output will obviously have D as the leaves of T. In conclusion, we will tacitly assume that we have access to the tree T in the following.

5.1 Leaf metrics: A duality approach

The μ-Edge Disjoint Packing problem asks to find an edge-disjoint packing $\mathcal{P}$ of paths whose endpoints both lie in a terminal set $S \subseteq V(G)$ that maximizes $\mathrm{pack}(\mu, G, S) := \sum_{P \in \mathcal{P}} \mu(s_P, t_P)$ (where s_P, t_P denote start- and endpoint of P). Hirai and Pap [10] show that if μ is a tree metric then $\mathrm{pack}(\mu, G, S) = \min_\lambda \max_{F \subseteq E} \sum_{uv \in E \setminus F} \mu(\lambda(u), \lambda(v))$, where λ is a zero-extension of the terminal-set S and the sets $F \subseteq E$ are edges whose deletion leaves every non-terminal vertex with an even degree. It follows that the maximum value of a half-integral τ-path packing is just the minimum cost of a τ-extension λ, since a half-integral path-packing is just a path-packing in the graph where every edge of G has been duplicated.

Let in the following $I = (G, \tau, \mu, q)$ be an instance of Zero Leaf Extension, where μ is a leaf metric over a tree T with leaves D. Let $\hat{\mu} = \mathrm{d}_T$ be the underlying tree metric. We define the *relaxed instance* $\hat{I} = (G, \tau, \hat{\mu}, q)$. Let $\mathbf{opt}(I)$, $\mathbf{opt}(\hat{I})$ denote the set of optimal solutions for the integral and the relaxed instance, respectively. Using this notation, we can summarize the duality: Given a relaxed instance $\hat{I}$, there exists a half-integral τ-path-packing $\mathcal{P}$ of cost precisely $\mathbf{cost}(\hat{I})$. In the following we will assume, by the usual identification argument, that τ is a bijection and a τ-path packing is equivalent to an S-path packing.

▶ **Lemma 13** (⋆). *Let $\mathcal{P}$ be an half-integral τ-path packing with $\frac{1}{2} \sum_{P \in \mathcal{P}} \mu(\tau(s_P), \tau(t_P)) = \mathbf{cost}(\hat{I})$. Let $\lambda \in \mathbf{opt}(\hat{I})$ be a relaxed optimum and let $P \in \mathcal{P}$ with endpoints s, t. Then $\mathbf{cost}_{\hat{\mu}}(\lambda, P) = \mu(\tau(s), \tau(t))$.*

A direct consequence is that if we trace an s-t-path $P \in \mathcal{P}$, then the labels assigned by any relaxed optimum λ to P induce a monotone sequence from s to t in T. That is, not only will we only encounter those labels that lie on $T[s,t]$, we also will encounter them 'in order'.

Consider an edge $xy \in E(T)$. Then, as a consequence of Lemma 13 the set of edges $C_{xy}(\lambda) = \{uv \in E(G) \mid \lambda(u) \in T_x, \lambda(v) \in T_y\}$ between the vertex sets with labels in T_x and T_y, respectively, must be saturated by paths of the packing $\mathcal{P}$. For cuts right above leafs of T, this implies the following.

▶ **Lemma 14** ($\star$)**.** *Let S be the vertices labelled by τ in G and assume that τ is a bijection. Let C be any minimum $(x, S-x)$-cut for some terminal $x \in S$. Then every optimal, half-integral S-path-packing in G will saturate C.*

▶ **Theorem 4.** *Let $I = (G, \tau, \mu, q)$ be an instance of* Zero Extension *where μ is a leaf metric on a set of labels D in a tree T, and let $\hat{I} = (G, \tau, \hat{\mu}, q)$ be the relaxed instance. Let $p = \mathbf{cost}(\hat{I})$. Then we can solve I in time $O(|D|^{q-p}|T|m + |T|\phi(n,m))$, where ϕ is the time needed to run a max-flow algorithm.*

Proof sketch. We first construct for every edge $ij \in T$ a flow network H_{ij} from G as follows: let D_i be those leaves that lie in the same component as i in $T - ij$ and D_j all others. Then H_{ij} is obtained from G by identifying all terminals $\tau^{-1}(D_i)$ into a source s and all terminals $\tau^{-1}(D_j)$ into a sink t. For each H_{ij} we compute a maximum flow f_{ij} in time $\phi(n,m)$. We can show that for every $\lambda \in \mathbf{opt}(\hat{I})$ it holds that $\sum_{xy \in T} |f_{ij}| = \sum_{xy \in T} |C_{xy}(\lambda)| = \mathbf{cost}_\mu(\hat{I})$. Note that we can also, in linear time, find the furthest cuts $\mathrm{C}_{\max}(x)$ for terminals $x \in S$ using the residual network of (H_{ij}, f_{ij}) with $i = \tau(x)$ and j the parent of i in T.

Next, we test whether G contains a vertex u that is not part of a furthest min-cut $\mathrm{C}_{\max}(x)$ for any $x \in S$; such a vertex cannot take an integral value in any relaxed optimum. We then branch on the $|D|$ possible integral values for u: for $x \in S$ with $\tau(x) \in D$ being the chosen integral value, we update the networks (H_{ij}, f_{ij}) by adding an edge xu of infinite capacity, then augment the flow. The number of augmentations is $\leqslant k - p$, as each augmentation witnesses the increase of p and thus the decrease of the parameter. We charge each augmentation to a level of the search tree and thus spend only $O(m)$ time per flow f_{ij}, for a total of $O(|T|m)$.

Otherwise, we find that every vertex of the current graph G is contained in at least one furthest min-cut. It can be shown (see full version) that the intersection of three or more such cuts is empty. Consequently, the graph decomposes into sets whose label is either fixed or is one of two possible values. A simple cut argument shows that we can fix one of the two labels greedily for the latter, and we construct an integral solution that matches the relaxed optimum. ◀

5.2 VCSP toolkit

Given a set of cost functions Γ over a domain D, an instance I of VCSP(Γ) is defined by a set of variables V and a sum of valued constraints $f_i(\bar{v}_i)$, where for each i, $f_i \in \Gamma$ and $\bar{v}_i$ is a tuple of variables over V. We write $f_i(\bar{v}) \in I$ to signify that $f_i(\bar{v})$ is a valued constraint in I. It is known that the tractability of a VCSP is characterized by certain algebraic properties of the set of cost functions. In full generality, such conditions are known as *fractional polymorphisms* for the finite-valued case and more general *weighted polymorphisms* in the general-valued case. Dichotomies are known in these terms both for the finite-valued [30] and general case of VCSP [16], i.e., characterizations of each VCSP as being either in P or NP-hard. We will only need a less general term.

A *binary multimorphism* $\langle \circ, \bullet \rangle$ of a language Γ over a domain D is a pair of binary operators that satisfy $f(\bar{x}) + f(\bar{y}) \geqslant f(\bar{x} \circ \bar{y}) + f(\bar{x} \bullet \bar{y})$, for all $f \in \Gamma, \bar{x}, \bar{y} \in D^{\mathbf{ar}(f)}$, where

$\mathbf{ar}(f)$ is the arity of f and where we extend the binary operators to vectors by applying them coordinate-wise. An operator $\circ$ is *idempotent* if $x \circ x = x$ for every $x \in D$, and *commutative* if $x \circ y = y \circ x$. A (finite, finite-valued) language Γ with a binary multimorphism where both operators are idempotent and commutative is solvable in polynomial time via an LP-relaxation [30]. The most basic example is the Boolean domain $D = \{0, 1\}$, in which case the multimorphism $\langle \wedge, \vee \rangle$ corresponds to the well-known class of *submodular functions*, which is a tractable class that generalizes cut functions in graphs.

The following is folklore, but will be important to our investigations. Again, the corresponding statements apply for arbitrary fractional polymorphisms, but we only give the version we need in the present paper.

▶ **Definition 15** (Preserved under equality)**.** Let f be a function that admits a multimorphism $\langle \circ, \bullet \rangle$. We say that two tuples $\bar{x}, \bar{y} \in D^{\mathbf{ar}(f)}$ are *preserved under equality* if $f(\bar{x}) + f(\bar{y}) = f(\bar{x} \circ \bar{y}) + f(\bar{x} \bullet \bar{y})$. For a relation $R \subseteq D^{\mathbf{ar}(r)}$, we say that f is *preserved under equality in* R if every pair of tuples $\bar{x}, \bar{y} \in R$ is preserved under equality and $\bar{x} \circ \bar{y}, \bar{x} \bullet \bar{y} \in R$.

▶ **Lemma 16** ($\star$)**.** *Let Γ be a language of cost functions that admit a multimorphism $\langle \circ, \bullet \rangle$ and let $\lambda_1, \lambda_2 \in \mathbf{opt}(I)$ for some instance I of* VSCP(Γ)*. Then for every valued constraint $f(\bar{v}) \in I$ it holds that $f(\lambda_1(\bar{v})) + f(\lambda_2(\bar{v})) = f((\lambda_1 \circ \lambda_2)(\bar{v})) + f((\lambda_1 \bullet \lambda_2)(\bar{v}))$, where $f(\lambda(\bar{v})) = f(\lambda(v_1), \ldots, \lambda(v_r))$ for $\bar{v} = v_1, \ldots, v_r$ is the value of $f(\bar{v})$ under λ. In other words, every valued constraint $f(\bar{v}) \in I$ is preserved under equality in $\mathbf{opt}(I)$.*

To illustrate, let us return again to the case of graph cut functions and submodularity over the Boolean domain. Let $G = (V, E)$ be an undirected graph, and define the cut function $f_G \colon 2^V \to \mathbb{Z}$ as $f_G(S) = |\delta(S)|$. Then f_G is the sum over binary valued constraints $f(u, v) = [u \neq v]$ over all edges $uv \in E$, in Iverson bracket notation. Since a single valued constraint $f(u, v)$ is submodular, the same holds for the cut function as a whole. Then Lemma 16 specialises into the statement that for two sets $A, B \subset V$ such that $\delta(A), \delta(B)$ are minimum s-t-cuts in G for some $s, t \in V$, there is no edge between $A \setminus B$ and $B \setminus A$. This kind of observation is a common tool in, e.g., graph theory and approximation algorithms.

The above lemma will be very useful when reasoning about the structure of $\mathbf{opt}(I)$ subject to more complex multimorphisms, as we will define next.

5.3 Submodularity on trees

Let $\preceq_T$ denote the ancestor relationship in a rooted tree T. For a path $P[x, y] \subseteq T$, let z_1, z_2 be the middle vertices of $P[x, y]$ (allowing $z_1 = z_2$ in case $P[x, y]$ has odd length) such that $z_1 \preceq_T z_2$. Define the commutative operators $\curvearrowleft, \curvearrowright$ as returning exactly those two mid vertices, e.g. $x \curvearrowleft y = y \curvearrowleft x = z_1$ and $x \curvearrowright y = y \curvearrowright x = z_2$. Languages admitting the multimorphism $\langle \curvearrowleft, \curvearrowright \rangle$ are called *strongly tree-submodular.*

Define the commutative operator $\uparrow$ to return the common ancestor of two nodes x, y in a rooted tree T. Define $x \nearrow y$ to be the vertex z on $P[x, y]$ which satisfies $\mathrm{d}_T(x, z) = \mathrm{d}_T(y, x \uparrow y)$. In other words, to find $z = x \nearrow y$, we measure the distance from y to the common ancestor of x and y and walk the same distance from x along $P[x, y]$. Languages that admit $\langle \uparrow, \nearrow \rangle$ as a multimorphism are called *weakly tree-submodular.* In particular, all strongly tree-submodular languages are weakly tree-submodular [15]. Tree-metric are, not very surprisingly, strongly tree-submodular:

▶ **Lemma 17** ($\star$)**.** *Every tree-metric is strongly (and thus also weakly) tree-submodular for every rooted version of the tree.*

We will need the following characterization of which value-pairs are preserved under equality by strong tree submodularity for tree distance functions. The proof follows from a long case analysis which we omit here.

▶ **Lemma 18** ($\star$). *Two tuples $(a, b), (x, y) \in V(T) \times V(T)$ are preserved under equality by d_T with multimorphism $\langle \curvearrowleft, \curvearrowright \rangle$ iff all four nodes lie on a single path P in T and either $a, b \leqslant_P x, y$ or $a, x \leqslant_P b, y$.*

▶ **Corollary 19** ($\star$). *Let d_T be preserved under equality in R for some $R \subseteq V_T \times V_T$, with at least one pair $(a, b) \in R$ with $a \neq b$. Then there is a path P in T which can be oriented as a directed path such that for every pair $(a, b) \in R$ the nodes a and b lie on P with $a \preceq_P b$.*

5.4 The domain consistency property

Consider a problem VCSP(Γ) over a domain D_I and a discrete relaxation VCSP(Γ') of VCSP(Γ) over a domain $D \supseteq D_I$. We say that the relaxation has the *domain consistency property* if the following holds: for any instance I of VCSP(Γ'), if for every variable v there is an optimal solution to I where v takes a value in D_I, then there is an optimal solution where all variables take values in D_I, i.e. an optimal solution to the original problem at the same cost. We show that the discrete relaxations of Zero Extension and Metric Labelling on induced tree metrics have the domain consistency property, allowing for FPT algorithms under the gap parameter via simple branching algorithms.

The result builds on a careful investigation of the binary constraints that $\mathbf{opt}(I)$ can induce on a pair of vertices $u, v \in V$, starting from Corollary 19. For the rest of the section, let us fix a relaxed instance $I = (G = (V, E), \tau, \mu, q)$ of Zero Extension where μ is a tree metric defined by a tree T, and the original (non-relaxed) metric is the restriction of μ to a set of nodes D_I. Note that I can be expressed as a VCSP instance using assignments and the cost function μ. Let $\mathbf{opt}$ be the set of optimal labellings. For a vertex $v \in V$, let $D(v)$ denote the set $\{\lambda(v) \mid \lambda \in \mathbf{opt}\}$, and let $D_I(v) = D_I \cap D(v)$. Furthermore, for a pair of vertices $u, v \in V$, let $R(u, v) = \{(\lambda(u), \lambda(v)) \mid \lambda \in \mathbf{opt}\}$ be the projection of $\mathbf{opt}$ onto (u, v), and $R_I(u, v) = R(u, v) \cap (D_I \times D_I)$ the integral part of this projection. We begin by observing that the "path property" of Corollary 19 applies to all vertices and edges in $\mathbf{opt}$.

▶ **Lemma 20** ($\star$). *For every vertex v that lies in a connected component of G containing at least one terminal, $D(v)$ is a path in T. Furthermore, for every edge $uv \in E$, $R(u, v)$ embeds into the transitive closure of a directed path in T.*

Next, we show the main result of this section: if u and v is a pair of variables, then whether or not there is an edge uv in E, the constraint $R(u, v)$ induced on u and v by $\mathbf{opt}$ is only non-trivial on values in $D(u) \cap D(v)$.

▶ **Lemma 21** ($\star$). *Let u and v be a pair of variables and $a \in D(u)$, $b \in D(v)$ a pair of values. If $(a, b) \notin R(u, v)$, then $a, b \in D(u) \cap D(v)$ and $a \neq b$.*

This gives us the following algorithmic consequence.

▶ **Lemma 22.** *There is a labelling $\lambda \in \mathbf{opt}$ such that for every variable v with $D_I(v)$ non-empty, we have $\lambda(v) \in D_I$.*

Proof sketch. Order $V(T)$ such that the values D_I come first, and assign to every variable v the value of $D(v)$ that is earliest in this ordering. By Lemma 21, this gives an assignment that is consistent with $R(u, v)$ for every pair of variables $u, v \in V$, and since $\mathbf{opt}$ has a majority polymorphism, this implies that the resulting assignment is in $\mathbf{opt}$. ◀

Let us for reusability spell out the explicit assumptions and requirements made until now.

▶ **Theorem 23** ($\star$). *Let $I = (G = (V, E), \tau, \mu, q)$ be an instance of* Zero Extension *with no isolated vertices and where every connected component of G contains at least two terminals, and where μ is an induced tree metric for some tree T and integral nodes $D_I \subseteq T$. Additionally, assume a collection of cost functions $\mathcal{F} = (f_i(\bar{v}_i))_{i=1}^m$ has been given, where for every f_i the scope is contained in V and where f_i is weakly tree submodular for every rooted version of T. Let I' be the VCSP instance created from the sum of the cost functions of I and $\mathcal{F}$. Then I' has the domain consistency property, i.e., there is an integral relaxed optimum if and only if every vertex v is integral in at least one relaxed optimum of I'.*

5.5 Gap algorithms for general induced tree metrics

By Theorem 23, we get FPT algorithms parameterized by the gap parameter $q - p$.

▶ **Theorem 3.** *Let $I = (G, \tau, \mu, q)$ be an instance of* Zero Extension *where μ is an induced tree metric on a set of labels D in a tree T, and let $\hat{I} = (G, \tau, \hat{\mu}, q)$ be the relaxed instance. Let $p = \mathbf{cost}(\hat{I})$. Then we can solve I in time $O(|D|^{q-p}|T||D|nm)$.*

Proof. This algorithm is similar to the algorithm for a leaf metric, except that we are not as easily able to test whether every variable has an integral value in **opt**. By the results of Section 5.1, the value of **opt** is witnessed by the collection of min-cuts for edges in T; we will use this as a value oracle for I. We initially compute a max-flow across every edge of T, then for every assignment made we can compute the new value of **opt** using $O(|T|)$ calls to augmenting path algorithms. This allows us to test for optimality of an assignment in $O(|T|m)$ time. The branching step then in general iterates over at most n variables, testing at most $|D|$ assigned values for each, and testing for optimality each time. Hence the local work in a single node of the branching tree is $O(|T||D|nm)$. This either produces a variable for branching on or (by Theorem 23) produces an integral assignment, and in each branching step the value of p increases but q does not. The time for the initial max-flow computation is eaten by the factor $|T|nm$. The result follows. ◀

For Metric Labelling, we first need to restrict the unary cost functions. Intuitively, the property is analogous to linear interpolation or convexity, applied along paths of the tree. The precise definition is as follows.

▶ **Lemma 24** ($\star$). *Let $f \colon V(T) \to \mathbb{R}$ be a unary function on a tree T. Then f is weakly tree submodular on T for every choice of root $r \in V_T$ if and only if it observes the following* interpolation property*: for any nodes $u, v \in V(T)$, at distance $\mathrm{d}_T(u, v) = d$, and every $i \in [d-1]$, let w_i be the node on $T[u, v]$ satisfying $d_T(u, w_i) = i$. Then for any such choice of u, v and i, it holds that $f(w_i) \leqslant ((d-i)/d)f(u) + (i/d)f(v)$.*

Let $f_0 \colon U \to \mathbb{Z}^+$ be a non-negative function defined on a subset U of the nodes of a tree T. We say that f_0 *admits an interpolation on* T if there is an extension $f \colon V(T) \to \mathbb{Z}^+$ with the interpolation property such that $f(v) = f_0(v)$ for every $v \in U$. Note that this only restricts the values $f_0(u)$ for nodes $u \in U$ that lie on a tree path between two other nodes $u_1, u_2 \in U$. In particular, if U is the set of leaves of T, then every function f_0 admits an interpolation by simply padding with zero values (although stronger interpolations are in general both possible and desirable).

We get the following.

▶ **Theorem 5.** *Let $I = (G, \sigma, \mu, q)$ be an instance of* Metric Labelling *where μ is an induced tree metric for a tree T and a set of nodes $D \subseteq V(T)$, and where every unary cost $\sigma(v, \cdot)$ admits an interpolation on T. Let $\hat{I} = (G, \hat{\sigma}, \hat{\mu}, q)$ be the relaxed instance, and let $p = \mathbf{cost}(\hat{I})$. Then the instance I can be solved in time $O^*(|D|^{q-p})$. In particular, this applies for any σ if D is the set of leaves of T.*

Proof. Assume that G is connected, or else repeat the below for every connected component of G. Select two arbitrary vertices $u, v \in V$ and exhaustively guess their labels; in the case that you guess them to have the same label, identify the vertices in I (adding up their costs in σ) and select a new pair to guess on. Note that this takes at most $O(|D|^2 n)$ time, terminating whenever you have guessed more than one label in a branch or when you have guessed that all vertices are to be identical. This guessing phase can only increase the value of p. We may now treat u and v as terminals, and the instance I as the sum of a Zero Extension instance on those two terminals and a collection of additional unary cost functions $\sigma(v', \cdot)$, as in Theorem 23. Note that the resulting VCSP is tractable, i.e., the value of an optimal solution can be computed in polynomial time. The running time from this point on consists of iterating through all variables verifying whether each one has an integral value in some optimal assignment, and branching exhaustively on its value if not. ◀

In particular, as noted, for a leaf metric μ the algorithm applies without any assumptions on σ (and without T being explicitly provided).

6 Conclusion

We have given a range of algorithmic results for the Zero Extension and Metric Labelling problems from a perspective of parameterized complexity. Most generally, we showed that Zero Extension is FPT parameterized by the number of *crossing edges* of an optimal solution, i.e. the number of edges whose endpoints receive distinct labels, for a very general class of cost functions μ that need not even be metrics. This is a relatively straight-forward application of the technique of recursive understanding [3].

For the case that μ is a metric we gave two stronger results for the same parameter. First, we showed a linear-time FPT algorithm, with a better parameter dependency, using an important separators-based algorithm. Second, and highly surprisingly, we show that every graph G with a terminal set S admits a polynomial-time computable, polynomial-sized *metric sparsifier* G', with $O(k^{s+1})$ edges, such that (G', S) mimics the behaviour of (G, S) over *any* metric on at most s labels (up to solutions with crossing number k). This is a direct and seemingly far-reaching generalization of the polynomial kernel for s-Multiway Cut [17], which corresponds to the special case of the uniform metric.

Finally, we further developed the toolkit of discrete relaxations to design FPT algorithms under a *gap parameter* for Zero Extension and Metric Labelling where the metric is an induced tree metric. This in particular involves a more general FPT algorithm approach, supported by an applicability condition of *domain consistency*, relaxing the previously used persistence condition.

Let us highlight some questions. First, is there a lower bound on the size of a metric sparsifier for s labels for Zero Extension? This is particularly relevant since the existence of a polynomial kernel for s-Multiway Cut whose degree does not scale with s is an important open problem, and since the metric sparsifier is a more general result.

Second, can the FPT algorithms for induced tree metrics parameterized by the relaxation gap be generalised to restrictions of other tractable metrics, such as graph metrics for median graphs or the most general tractable class of orientable modular graphs [9]? Complementing this, what are the strongest possible gap parameters that allow FPT algorithms for metrics that are either arbitrary, or do not explicitly provide their relaxation?

References

1 J. Besag. On the statistical analysis of dirty pictures. *J. Royal Stat. Soc. Ser. B*, pages 259–302, 1986.
2 J. Chen, Y. Liu, and S. Lu. An improved parameterized algorithm for the minimum node multiway cut problem. *Algorithmica*, 55(1):1–13, 2009.
3 R. Chitnis, M. Cygan, M. Hajiaghayi, M. Pilipczuk, and M. Pilipczuk. Designing FPT algorithms for cut problems using randomized contractions. *SIAM J. Sci. Comput.*, 45(4):1171–1229, 2016.
4 J. Chuzhoy and J. Naor. The hardness of metric labeling. *SIAM J. Sci. Comput.*, 36(5):1376–1386, 2007.
5 J.C. Culberson and P. Rudnicki. A fast algorithm for constructing trees from distance matrices. *Inf. Process. Lett.*, 30(4):215–220, 1989.
6 J. Fakcharoenphol, C. Harrelson, S. Rao, and K. Talwar. An improved approximation algorithm for the 0-extension problem. In *Proc. of 14th SODA*, pages 257–265. SIAM, 2003.
7 J. Fakcharoenphol, S. Rao, and K. Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *J. Comput. Syst. Sci.*, 69(3):485–497, 2004.
8 H. Hirai. The maximum multiflow problems with bounded fractionality. *Math. Oper. Res.*, 39(1):60–104, 2014.
9 H. Hirai. Discrete convexity and polynomial solvability in minimum 0-extension problems. *Math. Program.*, 155(1-2):1–55, 2016.
10 H. Hirai and G. Pap. Tree metrics and edge-disjoint S-paths. *Math. Program.*, 147(1-2):81–123, 2014.
11 Y. Iwata, M. Wahlström, and Y. Yoshida. Half-integrality, LP-branching, and FPT algorithms. *SIAM J. Comput.*, 45(4):1377–1411, 2016.
12 Y. Iwata, Y. Yamaguchi, and Y. Yoshida. Linear-time FPT algorithms via half-integral non-returning A-path packing. *CoRR*, abs/1704.02700, 2017.
13 H. Karloff, S. Khot, A. Mehta, and Y. Rabani. On earthmover distance, metric labeling, and 0-extension. *SIAM J. Sci. Comput.*, 39(2):371–387, 2009.
14 A.V. Karzanov. Minimum 0-extensions of graph metrics. *Eur. J. Comb.*, 19(1):71–101, 1998.
15 V. Kolmogorov. Submodularity on a tree: Unifying $L^\natural$natural-convex and bisubmodular functions. In *Proc. of 36th MFCS*, pages 400–411. Springer, 2011.
16 V. Kolmogorov, A. A. Krokhin, and M. Rolínek. The complexity of general-valued CSPs. *SIAM J. Comput.*, 46(3):1087–1110, 2017.
17 S. Kratsch and M. Wahlström. Representative sets and irrelevant vertices: New tools for kernelization. In *Proc. of 53rd FOCS*, pages 450–459. IEEE Computer Society, 2012.
18 L. Lovász. Flats in matroids and geometric graphs. In *Proc. of 6th Br. Comb. Conf.*, Combinatorial Surveys, pages 45–86, 1977.
19 R. Manokaran, J. Naor, P. Raghavendra, and R. Schwartz. SDP gaps and UGC hardness for multiway cut, 0-extension, and metric labeling. In *Proc. of 40th STOC*, pages 11–20, 2008.
20 D. Marx. Parameterized graph separation problems. *Theor. Comput. Sci.*, 351(3):394–406, 2006.
21 D. Marx. A parameterized view on matroid optimization problems. *Theor. Comput. Sci.*, 410(44):4471–4479, 2009.

22 D. Marx and I. Razgon. Fixed-parameter tractability of multicut parameterized by the size of the cutset. *SIAM J. Comput.*, 43(2):355–388, 2014.

23 M. Naor, L.J.Schulman, and A. Srinivasan. Splitters and near-optimal derandomization. In *Proc. of 36th FOCS*, pages 182–191. IEEE, 1995.

24 M. Óskarsdóttir, C. Bravo, W. Verbeke, C. Sarraute, B. Baesens, and J. Vanthienen. A comparative study of social network classifiers for predicting churn in the telecommunication industry. In *Proc. of 8th ASONAM*, pages 1151–1158. IEEE, 2016.

25 B. Pang and L. Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proc. of 43rd ACL*, pages 115–124. Association for Computational Linguistics, 2005.

26 J. Picard and H.D. Ratliff. A cut approach to the rectilinear distance facility location problem. *Oper. Res.*, 26(3):422–433, 1978.

27 Felix Reidl and Magnus Wahlström. Parameterized algorithms for zero extension and metric labelling problems. *CoRR*, abs/1802.06026, 2018. `arXiv:1802.06026`.

28 A. Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency.* Springer, 2003.

29 J. Kleinberg and E. Tardos. Approximation algorithms for classification problems with pairwise relationships: Metric labeling and markov random fields. *J. ACM*, 49(5):616–639, 2002.

30 J. Thapper and S. Živný. The complexity of finite-valued CSPs. *J. ACM*, 63(4):37, 2016.

31 N. Vikas. A complete and equal computational complexity classification of compaction and retraction to all graphs with at most four vertices and some general results. *J. Comput. Syst. Sci.*, 71(4):406–439, 2005.

32 M. Wahlström. LP-branching algorithms based on biased graphs. In *Proc. of 28th SODA*, pages 1559–1570. SIAM, 2017.